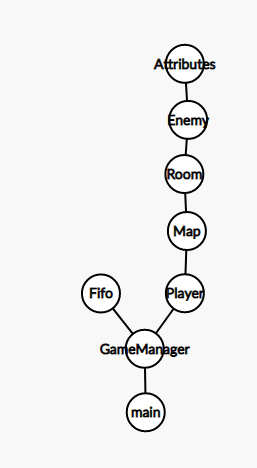
**Knight Runners**

**Dokumentacja dla programisty**

**Struktura gry** – opisanie z jakiego modułu dany moduł dziedziczy.



**Attributes**Moduł trzymający tylko strukturę atrybutów w której trzymane są liczby całkowite odpowiadające ***Sile***, **Zręczności**, **Inteligencji**. Trzymana jest również funkcja ***Create\_Attributes*** do tworzenia i ustawienia tej struktury.

**Enemy**Moduł trzymający strukturę przeciwnika. W tej strukturze jest trzymany:

- ***EnemyID***, czyli jaki to jest przeciwnik, (1 – przeciwnik na siłę, 2 – przeciwnik na zręczność, 3 – przeciwnik na inteligencję, 4 – biblioteka, 5 – pokój z klucznikiem)

- ***Attributes***,czyli atrybuty przeciwnika,

- ***Defeated*** zmienna boolowska czy został pokonany.

Trzymana jest również funkcja ***Create\_Enemy*** do tworzenia i ustawienia tej struktury o odpowiednie wartości.

**Room**

Moduł trzymający strukturę pokoju. W skład tej struktury wchodzą następujące rzeczy:

- ***ID,*** czyli jaki to jest typ pokoju (0 – pusty, 1 – ściana, 2 – gracz, 3 – pokój z klucznikiem, 5 – księżniczka)

- ***Enemy*** – jakiego przeciwnika możemy tutaj spotkać.

Trzymana jest również funkcja ***Create\_Room*** do tworzenia i ustawienia tej struktury.

**Map**

Moduł trzyma dwie tablice 2 wymiarowe potrzebne to zapamiętania odpowiednio charów oraz GTKWidgetów w gridzie danej mapy.

Ponadto trzymana jest pozycja startowa gracza i rozmiary mapy. Są dodatkowo dwie funkcje z czego Jedna ***Create\_Map\_GTK*** tworzy mapę na gridzie, a druga ***Get\_Map*** ma za zadanie wczytać jedną z wcześniej wylosowanych map o podanym ID.

Co do ***Get\_Map*** wczytuje jedną mapę z pliku, która wygląda następująco  
**MapSizeX,MapSizeY  
PlayerPosX, PlayerPosY  
plansza charów o rozmiarze MapSizeX, MapSizeY  
Ile Specjalnych Pokoi**I każdy pokój opisany jest **PosRoomX, PosRoomY, EnemyID, EnemyStrength, EnemyAgility, EnemyInteligence.**

**Player**

Moduł trzymający strukturę gracza. W skład tej struktury wchodzi:

- ***x,y*** – obecna pozycja gracza,

- ***prevX, prevY*** – poprzednia pozycja gracza,

- ***HP*** – Obecna ilość HP gracza,

- ***WhereStanding*** – Na jakie pole gracz stanął,

- ***Display*** – Co ma pokazywać gracz,

- ***Attributes*** – Atrybuty gracza,

Trzymana jest również funkcja ***Create\_Player*** do tworzenia i ustawienia tej struktury o zadane parametry.

**Fifo**

Moduł do stworzenia plików kolejkowych, który umożliwia przesyłanie informacji między dwoma plikami oraz odczytywania z nich.

**GameManager**

Moduł który trzyma odpowiednio wszystkie labele związane z UI, które się zmieniają podczas gry. Oprócz tego w pliku powstały odpowiednie kategorie:

- **Fifo** – czyli wszystko potrzebne do przekazania informacji czy gracz wykonał turę.

a) ***Send\_Turn\_Info\_To\_Pipe*** – wysyła informacje do potoku, że dany gracz zrobił ruch w swojej turze

b) ***Get\_Turn\_Info\_From\_Pipe*** – sprawdza czy przypadkiem drugi gracz nie wykonał już ruchu.

c) ***Pick\_Map*** – losuje jedną z 3 map na której będzie rozgrywać się gra i przekazuje drugiemu graczowi potokiem na jakiej mapie odbędzie się potyczka.

- **Turn Managment** – tablica, jaka tura się wykonała oraz jaka jest obecnie Tura.

- **GtkStuff** – funkcje pokazujące komunikaty z gtk

a) ***Show\_Error*** – pokazuje DialogBoxa z błędem

b) ***Show\_Communicate*** – pokazuje komunikat o zadanej wiadomości

c) ***Show\_Room\_Communicate*** – pokazuje komunikat związany z wejściem do danego typu pokoju

d) ***End\_Game*** – przekazuje komunikat o zakończonej grze oraz wychodzi z gry po przeczytaniu komunikatu

e) ***Update\_Stats*** – pokazuje komunikat, która statystyka się zwiększyła

- **Start Game or Updating**

a) ***Init\_Game*** – przed rozpoczęciem gry ustawia wszystkie wartości zmiennych na to na co powinny być

b) ***Update\_Grid*** – aktualizuje pozycje gracza na mapie oraz sprawdza czy nie jest na polu z księżniczką, albo przeciwnikiem.

c) ***Update\_Attributes*** – zwiększa atrybuty

d) ***HP\_Loss*** – funkcja obniża hp gracza i aktualizuje HPLabel.

e) ***Changing\_Room*** – funkcja zmieniająca pokój względem wyboru gracza.

- **Interacting with rooms**

a) ***Room\_With\_Enemy*** – wyświetla odpowiedni komunikat po wejściu do komnaty oraz sprawdza co wybrał gracz (jedna z trzech opcji)

b) ***Search\_Room*** – wykonuje odpowiednią operację eksploracji pokoju w zależności od tego, gdzie znajduje się gracz

- **Functions for buttons** – funkcje, które zwracają odpowiedni response po zamknięciu okna dialogowego

**Main**

Tutaj zostało stworzone wszystko co potrzebne z GTK oraz tutaj jest zaimplementowany „game loop” za pomocą komendy g\_timeout\_add.